Hierarchical document signature: A specialized application of fuzzy signature for document computing

Sukanya Manna
School of Computer Science
The Australian National University
ACT 0200, Australia

B. Sumudu Udaya Mendis
School of Computer Science
The Australian National University
ACT 0200, Australia

Tom Gedeon
School of Computer Science
The Australian National University
ACT 0200, Australia

***Abstract*—We develop document computing procedures for the analysis of discourse structures within a document, represented by hierarchical document signatures. A signature is a string of data characterizing a certain case (e.g. characteristics of a sentence in case of a document). The place of the individual data is fixed within the string, it holds a local value semantics. Fuzzy granulation is a semantic background technique for all kinds of information which originates from human estimation or recorded by human valuation of numerical data. For analysis of such data the development of special procedures is suggested, different from the usual statistical methods. We used a form of fuzzy signature, called hierarchical document signature to modularize an unstructured document in a hierarchical manner, from Document level to sentence level, sentence level to attribute level and then to word level. We used occurrence of words as the information of the lowest module to find the similarity among the next higher module by aggregating the signature values giving sentence pair coherence.**

Keywords: fuzzy signatures, aggregation, fuzzy measure, document signature, sentence similarity, vector valued fuzzy set

I. Introduction

Every document has a certain hierarchical structure concerning the importance of the words or concepts occurring in it. A document consists of sentences, and sentences consists of words. These form a hierarchical structure among themselves. Exploiting this feature in a document enables us to semantically bind each of the modules based on their proximity as in discourse.

In discourse, both explicit and implicit devices signify links between sentences, between groups of sentences, and between elements within sentences, and in turn, carry additional elements of discourse semantics. We thus take Discourse Structure (DS) [1] broadly, to cover all aspects of the internal organizational structure of a discourse. DS subsumes notions such as segmentation, relations between segments (informational and intentional), anaphoric relations, modal subordination, discourse topic, thematic progression, etc. DS is more commonly exploited in the applications of computational linguistic and NLP concepts like rhetorical structure theory (RST) [2], cross document structure theory (CST) [3]. But here, we consider DS especially segmentation, finding similarities (any form) between sentences in a document propagating the information from the word level to the higher modules using fuzzy measures and aggregations. This is presented by hierarchical document signature (HDS) model. This approach is useful when it is necessary to find which sentence pairs in a document are highly related to each other and to which degree. Based on this feature, filtering of sentences in a document can be done, which can further be analyzed for generation of similar documents automatically.

The hierarchical document signature (HDS) is a special type of fuzzy signature (FS) [4] which is used for document analysis purpose. Fuzzy signatures can describe, compare and classify objects with complex structure and interdependent features. The hierarchical organization of fuzzy signatures express the structural complexity of a problem. The local preference relations among the hierarchies and sub-branches of a fuzzy signature can be used to approximate the global preference relation of a decision problem. We exploited this feature of FS to find the sentence level similarity of a document by propagating the information from the words level to document level. HDS is a modified form of vector valued fuzzy sets.

The vector valued fuzzy sets (VVFS) concept has been further generalized in [5] to introduce the fuzzy signature concept. Fuzzy signatures can model sparse and hierarchically correlated data with the help of hierarchically structured vector valued fuzzy sets and a set of not-necessarily homogenous and hierarchically organized aggregation functions. The set of aggregation functions map the different universes of discourse of the hierarchical fuzzy signature structure, from lower branches to the higher branches. We argue that these properties help fuzzy signatures to model problems similar to the nature of human comprehensible hierarchical approaches to problem solving. An important advantage of the fuzzy signature concept is that it can be used to compare degree of similarity or dissimilarity of two slightly different objects, which have the same fuzzy signature skeleton.

II. Hierarchical Document Signature

In this section we present the hierarchical document signature which is comprised of modularized components of a

978-1-4244-3597-5/09/$25.00 ©2009 IEEE

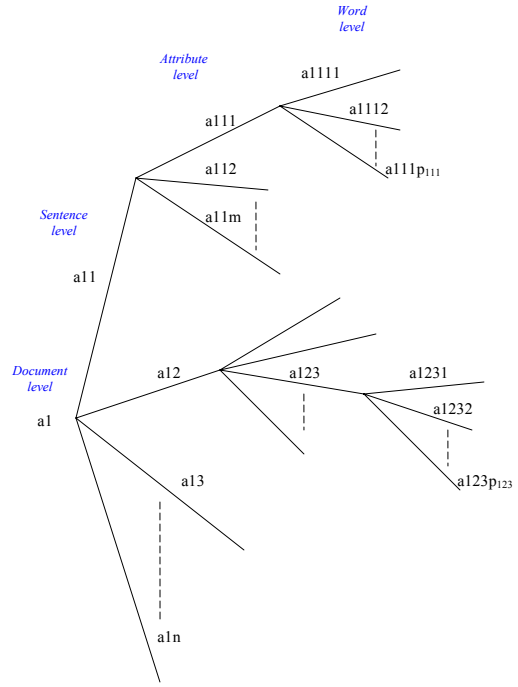

Fig. 1. Hierarchical Document Signature

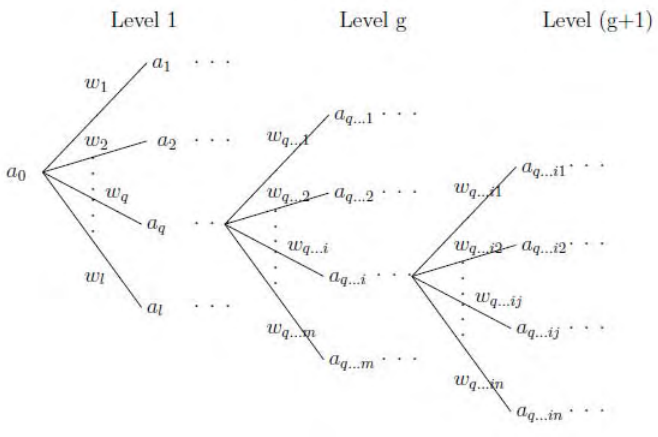

Fig. 2. Illustration of an arbitrary signature

document in signature form. Figure 1 illustrates a hierarchical document signature structure.

Definition 1: Fuzzy Signature is a VVFS, where each vector component is another VVFS (branch) or a atomic value (leaf), and denoted by,

$$A : X \rightarrow [a_i]_{i=1}^{k} \left(\equiv \prod_{i=1}^{k} a_i \right). \tag{1}$$

$$where\ a_i = \begin{cases} [a_{ij}]_{j=1}^{k_i} & ;if\ branch \\ [0,1] & ;if\ leaf \end{cases}$$

and Π describes the Cartesian product.

Lemma 1: Hierarchical Document Signature (HDS) can be defined as a special class of Fuzzy signature. In this application HDS structure is made up of four different kinds of vectors. These four vectors are document vector, sentence vector, attribute vector, and word vector; D_v, S_v, A_v and W_v.

$$d_i = [s_j] \in S_v;\ d_i \in D_v \tag{2}$$

$$s_j = [a_k] \in A_v;\ s_j \in S_v \tag{3}$$

$$a_k = [w_l] \in W_v;\ a_k \in A_v \tag{4}$$

In this paper we discuss more problem specific HDS. The levels and branches are flexible according to different applications. In fig.1, $a1$ represents a document at *document level*. A document is now segmented into n sentences, which we denote by $a11$ to $a1n$. This is the *sentence level* of the signature we developed. Next is the *attribute level*, which basically classifies the words of the sentences based on their attributes such as name, place, time etc. So, for each sentence, we have m different attributes, which is fixed for each document signature. In general m is a integer, but it is constant with specific application for the ease of comparison as shown in fig. 3. Each of these attributes per sentence is presented as $a111$ to $a11m$ for sentence 1, similarly, $a121$ to $a12m$ for the second sentence and so on. Now each attribute of each sentence has words, and this level is called *word level*. There can be any number of words in each attributes provided that those words must be present in the attribute files (attribute files are corresponding files for each attributes containing a list of attributes at the document level). Here, p_{111} is the maximum number of words for attribute $a111$. Suppose we have two signatures for sentence 1 and sentence 2 of a document. Now, for signature 1, if $m = 2$ (let the attributes be name and time), and for signature 2, $m = 4$ (let the attributes be name, time, place, term), then for two signatures it is tough to compare. It make it computationally very expensive, at the same time it loses practicality. As in this example, we can clearly see that there is no point of comparing attribute name with attribute place of the two signatures. It is meaningful to compare the same attributes.

a) Example of a sentence structure of HDS: Let us consider a sentence:
Sumudu and Amir will be leaving for a conference in Sydney at 6:30am. Now, according to our HDS model, in fig.3, we illustrate the corresponding sentence structure according the example. Figure 3 starts from the sentence level. *Sumudu* and *Amir* are the two names of persons found here, so they are placed under name attribute. *Sydney* is a place, so placed under place attribute, *conference* is a general term, so it is placed under term attribute. Likewise *6:30am* is a time present in the sentence, we placed in under the time attribute. In this model, we made attribute lis at the document level, which consists of the respective attributes found in a document. We use these lists to find those defined words in the attribute file in each sentence for our computation.

III. Sentence Similarity Using the Signature Model

We compute sentence similarity at the document level using HDS. At each of these levels, we used different fuzzy measures

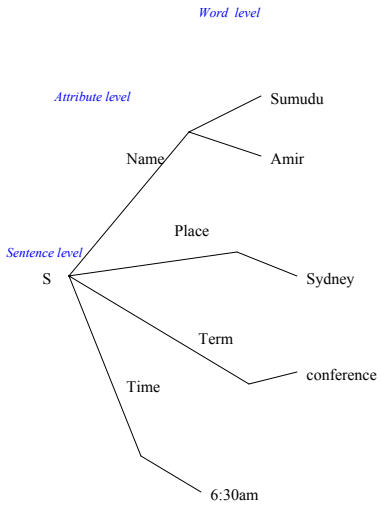

Fig. 3. Illustration of a sentence signature of HDS

and aggregations. In this section, we present them level by level.

A. Word level computations

In sec.II, we presented the HDS. At the word level, we store words per attributes. Along with the words, we keep the count of the number of occurrence of that corresponding word in that sentence. This is the frequency of that word.

b) Normalization of word frequency: Let D be a document vector, S ($|S| = n$) be the vector of sentences in D, and A ($|A| = m$) be the vector of attributes in S. W be the vector of words in attribute A. We have considered here four attributes, *name* (name of persons), *place*, *term* (significant terms of the document considered) and *time*. Let p, q, r, and s be the number of words of attributes found at the document level. W is the word vector at each attribute level such that $|W| \in p, q, r, s \in N$, where N is a natural number. $f_{w_1 a_{place} s_1}$ be the frequency of word w_1 for sentence s_1 and attribute a_{place}. Now at sentence level, for sentence s_z ($s_z \in S$), attribute a_y ($a_y \in A$), there can be word w_x We use a normalized frequency of words instead of the original word count per sentence. We calculate the normalized frequency ($nf_{w_x a_y s_z}$) of a word w_x in attribute a_y and sentence s_z, by

$$nf_{w_x a_y s_z} = \frac{f_{w_x a_y s_z}}{|s_z| - (|w_x|) \times f_{w_x a_y s_z}} \tag{5}$$

where $|s_z|$ is the length of the sentence s_z, and $|w_x|$ is the length of the word w_x respectively.

The main characteristic of classical measures theory is the additivity property. Many engineering applications were successfully designed with this property, but when it comes to soft computing applications the additivity property can be too rigid. The fuzzy measure is a generalization of the additive measures as it replace the additivity by the weaker condition of monotonicity [6].

Definition 2: A fuzzy measure on a discrete set $N = \{1, \ldots, n\}$ is a set function $v : 2^N \rightarrow [0, 1]$ that satisfy the following conditions:

(i) Boundary: $v(\emptyset) = 0$
(ii) Monotonicity: $A, B \subseteq N$ and $A \subseteq B$ then $v(A) \leq v(B)$

Now, we prove that normalization of word frequency [7] is a fuzzy measure in the context of this paper. The function nf (normalized frequency) will be a fuzzy measure, if it satisfies the condition and the function is monotonous [8]. To explain this, let us consider three words, A, B, and C. Now, also note that, in the context of this paper comparison of $nf(f_A, f_B)$ and $nf(f_A, f_C)$ is not meaningful, and we interested only comparison of $nf(f_A, f_B)$ and $nf(f_A, f_C)$, and $nf(f_A, f_B)$ and $nf(f_B, f_C)$.

1) Proof of property (i): $nf(f_\emptyset) = 0$, when frequency of word is 0 and thus according to equation (5) $nf(f_\emptyset, f_w) = 0$
2) Proof of property (ii): Let us assume, $\mid B \mid \leq \mid C \mid$, according to equation (5), we can write that $nf(f_A, f_B) \leq nf(f_A, f_C)$. monotonous

Thus, equation (5) stratifies all conditions to be a fuzzy measure.

c) Fuzzy model to calculate similarity between two words: After we find the normalized frequency of words, we need to calculate the similarity between two sentences in terms of these words at per attribute per sentence level. We calculate the similarity between two common words of same attribute of two different sentences.

Let w_x and w_u be two words for two different sentences s_z and s_v for attribute a_y. Similarity between these two words is presented by $sim(w_u, w_x)$. Now, in order to get the fuzzy value of the similarity function, according to [9] the following conditions must be satisfied:

1) if $nf(w_u) = nf(w_x)$, then $sim(w_u, w_x) = 1$
2) if $nf(w_u) \cap nf(w_x) = \emptyset$, then $sim(w_u, w_x) = 0$

Thus,

$$sim(w_u, w_x) = \frac{nf(w_u) \cap nf(w_x)}{nf(w_u) \cup nf(w_x)} \tag{6}$$

or

$$sim(w_u, w_x) = \frac{min[nf(w_u), nf(w_x)]}{max[nf(w_u), nf(w_x)]} \tag{7}$$

d) Generation of weights for attribute type: We assign specific weights for each attributes at the word level after calculating the fuzzy similarity between two words. We mentioned that we use separate attribute files for storing the words of each attributes encountered in the whole document. Let p, q, r, and s be the number of words of attributes found at the document level for *name* (name of persons), *place*, *term* (significant terms of the document considered) and *time*. Let wt_{name}, wt_{place}, wt_{term}, and wt_{time} be the corresponding weights for each of these four attributes. Thus, weights are

computed as:

$$wt_{place} = \frac{1}{q} \tag{8}$$

$$wt_{name} = \frac{1}{p} \tag{9}$$

$$wt_{term} = \frac{1}{r} \tag{10}$$

$$wt_{time} = \frac{1}{s} \tag{11}$$

So the final similarity for two words for a particular attribute is

$$sim_{weighted}(w_u, w_x) = sim(w_u, w_x) \times wt_y \tag{12}$$

where, $sim_{weighted}$ is the weighted similarity of two words w_u and w_x for attribute y, and wt_y is the weight of the attribute.

e) Mean aggregation of the weighted similarity values at attribute level: Now, for each attribute, we found the weighted similarity of the common words in two sentences. At this stage, we compute the mean of the weighted similarity values of all the words for that particular attribute. Now we propagate this mean weighted similarity from word level to attribute level. Let a_y be the attribute, $W_{t_{ia_y}}$ and $W_{t_{ja_y}}$ be the word vectors for a_y for sentences t_i and t_j such that $\{W_{t_{ia_y}}, W_{t_{ja_y}}\} \in W$ and $\{t_i, t_j\} \in S$. Now, $sim_{mean_{a_y}}$ be the mean weighted similarity of the words common in $W_{t_{ia_y}}$ and $W_{t_{ja_y}}$ word vectors for a_y. Thus,

$$sim_{mean_{a_y}}(|W_{t_{ia_y}} \cap W_{t_{ja_y}}|) = \frac{\sum sim_{weighted}(|W_{t_{ia_y}} \cap W_{t_{ja_y}}|)}{|W_{t_{ia_y}} \cap W_{t_{ja_y}}|} \tag{13}$$

f) Max aggregation at sentence level: At this level, we have the weighted similarity value for each attributes between a sentence pair which we calculated using eq.13. We now, find the maximum among these attribute values to get the overall degree of sentence similarity for that particular sentence. As we mentioned earlier in this section that we used four attributes, so for a pair of sentence $\{s_i, s_j\} \in S$, for attributes $\{a_{name}, a_{place}, a_{term}, a_{time}\} \in A$, we get the mean similarity for each attribute using eq.13. Now, we calculate the final similarity by taking max of these mean similarity for these attributes, which is

$$sim_{max} = max(sim_{mean_A}) \tag{14}$$

Eq.14, thus gives the overall degree of similarity between two sentences in the document. This value is then propagated into the sentence level giving the similarity between them.

IV. Implementation method and results

A. Data processing

We selected single documents for the HDS. For this, we created a small sample data set to explain the detailed feature of our method and we used a CST corpus [10] related to a Milan plane crash. This contains a number of documents related to the incident. The documents were tokenized, cleaned, and stemmed. The cleaning is done by removing the stop words and stemming is done only for the significant terms not for named entities which are extracted by named entity extractors.

g) Attribute and term selection: In this work, we classified the word level into attribute types to categorize them. We used NLP named entity extractor *LingPipe*[1] [11] and *SweNam*[2] [12] to extract names of person, places, time of events. We merged the output of these two tools and then manually rectified the entities extracted by them as sometimes these kind of NLP tools do not provide very accurate result if there is no training data set available.

The term attribute contains the significant terms in a document. We extracted the significant terms by a higher order SVD model in [13], which extracts significant topic oriented terms from a single document. We can see that, when four of these attributes are combined, it covers almost all the important information of a document.

B. Pseudocode of workflow

We assume below that the HDS skeleton is already created with the number of sentences present in the document and the four attributes mentioned in the attribute level of each sentence. We present here the computational steps only to get the sentence pair similarity.

Algorithm 1 Pseudocode of basic computational steps involved in HDS

```
Generate four attribute files (name, place, term, time)
Find words from attribute files for each sentence putting them into word level of the signature with their respective attributes
Count the number of occurrences (frequency) of these words at each sentence
Normalize the frequency as in eq.5
for Each sentence pair do
    for Each attribute type do
        Calculate sim_mean using eq.13
    end for
    Calculate sim_max using eq.14
end for
```

The sim_{max} calculated for each sentence pair gives the final similarity degree for them.

C. Results

We present sentence similarity using HDS. We used a small test document to illustrate the results, then we present the results on the original CST data set.

h) Results on a sample data: As shown in table I, we presented a short document of four sentences. For this we generated four attribute files, each for name, place, term and time respectively. The name file contains *Sukanya Sumudu Tom Amir*, place file contains *Sydney Canberra*, term file contains *dinner conference meeting* and finally the time file contains *5pm 6:30am*. Now, we stepwise implemented the method as shown in the pseudocode 1. The intermediate calculations are shown in sec.II.

[1] http://alias-i.com/lingpipe/web/demo-ne.html
[2] http://www.nada.kth.se/iplab/hlt/swenam/index-eng.html

TABLE I
SAMPLE DATA

1	Sumudu and Amir will be leaving for a conference at Sydney at 6:30am.
2	Sukanya will see Amir at Sydney airport for meeting and will have dinner.
3	Tom and Sukanya will leave Canberra to join Sumudu and Amir at Sydney for the conference and meeting.
4	A cat has bitten a dog in Canberra.

TABLE II
COMPARISON OF SENTENCE SIMILARITY WITH DIFFERENT AGGREGATION OPERATIONS AT WORD LEVEL

(s_i,s_j)	Max	Mean	Min	FJC
(1,2)	0.03	0.04	0.03	0.50
(1,3)	0.03	0.03	0.02	0.37
(1,4)	0.00	0.00	0.00	0.00
(2,3)	0.03	0.03	0.02	0.37
(2,4)	0.00	0.00	0.00	0.00
(3,4)	0.06	0.44	0.02	0.21

Table II, shows the comparison of similarity values of sentence pairs using eq.14 at sentence level, eq.13 at attribute level and used different aggregations at word level. For this work, we used fuzzy measure of Jaccard's coefficient (FJC as shown in the table) using eq.7. Now, instead of eq.7, we replaced it by equations 16, 17, and 15 for max, mean and min respectively. The first column of the table refers to the sentence pairs, second to fifth columns are the similarity values obtained by different aggregations used at the word level.

$$sim(w_u, w_x) = min[nf(w_u), nf(w_x)] \quad (15)$$

$$sim(w_u, w_x) = max[nf(w_u), nf(w_x)] \quad (16)$$

$$sim(w_u, w_x) = mean[nf(w_u), nf(w_x)] \quad (17)$$

If we compare the sentences given in table I with the results shown in table II, we can manually justify which pairs of sentences are most similar to each other. It is very clear that the aggregation of FJC in the word level shows very imprecise judgement similar to human understanding. In case of *min*, the similarity values for sentences (1,3), (2,3) and (3,4) are same. It is hard to understand which similarity is higher and more meaningful. Again, when we take *max*, we find similar problem with little difference in case. Here, the result is erroneous conceptually. As sentence (3,4) has only one word in common gives higher similarity value than (1,2) or (2,3), which have more similar words among them. *Mean* still performs better than the other two functions but not as good as FJC. Here, it still can mark out the similarity of sentence pairs reasonably. Thus we see that it works well when we take FJC for calculating the similarity.

In table III, we look at the effect of different aggregation at attribute level using FJC at word level and max at sentence level. This is an intermediate stage of the whole process. Using the sample data set, we found that aggregations of *max*, *mean*, and *min* to be same as number of common words for each attribute was either one or the weighted similarity at word level was same which gave same results for these attributes. For document computing, taking *mean* at this stage is more suitable than simple min or max as words in different sentence have different normalization effect. So, taking mean has the contribution of both rather than taking either min or max at this stage.

TABLE III
COMPARISON OF DIFFERENT AGGREGATIONS AT ATTRIBUTE LEVEL

(s_i,s_j)	Name	Place	Term	Time
(1,2)	max	0.04	0.03	-
	mean	0.04	0.03	-
	min	0.04	0.03	-
(1,3)	max	0.03	0.02	-
	mean	0.03	0.02	-
	min	0.03	0.02	-
(1,4)	max	-	-	-
	mean	-	-	-
	min	-	-	-
(2,3)	max	0.03	0.02	-
	mean	0.03	0.02	-
	min	0.03	0.02	-
(2,4)	max	-	-	-
	mean	-	-	-
	min	-	-	-
(3,4)	max	0.44	0.02	-
	mean	0.44	0.02	-
	min	0.44	0.02	-

TABLE IV
COMPARISON OF SENTENCE SIMILARITY WITH DIFFERENT AGGREGATION OPERATIONS AT SENTENCE LEVEL

(s_i,s_j)	Max	Mean	Min
(1,2)	0.5	0.38	0.25
(1,3)	0.37	0.27	0.18
(1,4)	0.00	0.00	0.00
(2,3)	0.37	0.27	0.18
(2,4)	0.00	0.00	0.00
(3,4)	0.21	0.21	0.21

In table IV, we present the final similarity value using a different aggregation at sentence level. For this part we use mean of table III and FJC from table II. The values for three of these aggregations are same for this data set used here. So, taking either of these can be reasonably good, we used the max for the final similarity value of sentences.

i) Similarity results with CST data: We present here sentence similarity of couple CST documents out of 9. The similarity results are based on the final similarity value we calculated eq.14.

Document 1'milan9 data set with file index 16': For this document, when we arranged the sentence pair similarity values in descending order we found, $(2,4) > (4,9) > (1,2) > (3,4)...(3,13) > (1,5)$.... For better understanding we present the sentences (1,2) having higher similarity than (1,5) (keeping a sentence common in both cases).

1 *ABCNEWS.com : Plane Slams Into Milan Skyscraper*
2 *The Pirelli Building in Milan, Italy, was hit by a small plane.*
5 *The weather was clear at the time of the crash.*

If we now consider these sentences, it is very obvious that the inference drawn by our method defines sentence similarity in a meaningful way.

Document 2 'milan9 data set with file index 19': In this case, like the previous case we again arrange the similarity values in descending order such that $(1,2) > (2,10) > (3,10) > (1,10) > ...(1,7) > (1,4) > (2,4)$.... We now present here few sentences which we presented in the form of sentence pair comparison.

1 *Small plane hits Milan building*
2 *Smoke rises from the Milan skyscraper*
4 *There were no immediate reports on casualties as rescue workers attempted to clear the area in the city s financial district.*
10 *Police and ambulances rushed to the building in downtown Milan*

Like the previous document, here we find meaningful sentence pair comparison as well. Sentence 4 seems to be different from the other three. We do not find any similarity with these and is ranked lower in the sentence pair list.

V. Conclusion

In this work we presented a hierarchical document (HDS) signature, which modularizes a document into hierarchical structure to find the similarity between sentences. This has a similar structure as fuzzy signature with predefined levels. For each document signature, the number of levels and number of branches in each level can vary based on the application. This model works well for finding sentence similarity in a document in an imprecise form more like human reasoning as shown in the results. We can also see that at word level, proper aggregation plays a very important role and it reflects the values at higher levels.

The importance of HDS is that we can easily compare inter document or intra document similarities either at sentence level or at document level. Therefore for our future work, we will use our signature in different types of document analysis problems.

References

[1] B. Grosz and C. Sidner, "Attention, intentions, and the structure of discourse," *Computational Linguistics*, vol. 12, no. 3, pp. 175–204, 1986.
[2] W. Mann, S. Thompson, B. Baldwin, T. Morton, A. Bagga, J. Baldridge, R. Chandraseker, A. Dimitriadis, K. Snyder, M. Wolska, *et al.*, "Rhetorical structure theory: Toward a functional theory of text organization," *Computers & Mathematics with Applications*, vol. 23, pp. 133–177, 1998.
[3] Z. Zhang, J. Otterbacher, and D. Radev, "Learning cross-document structural relationships using boosting," in *Proceedings of the twelfth international conference on Information and knowledge management*. ACM New York, NY, USA, 2003, pp. 124–130.
[4] B. Mendis and T. Gedeon, "Aggregation selection for hierarchical fuzzy signatures: A comparison of hierarchical owa and wrao," in *IPMU'08*, 2008.
[5] L. Kóczy, T. Vámos, and G. Biró, "Fuzzy signatures," in *EUROFUSE-SIC '99*, 1999, pp. 210–217.
[6] M. Sugeno and T. I. of Technology, *Theory of fuzzy integrals and its applications*. Tokyo Institute of Technology Tokyo, Japan, 1974.
[7] "Jaccard's similarity coefficient." [Online]. Available: http://en.wikipedia.org/wiki/Jaccard_similarity_coefficent
[8] M. Sugeno, "Fuzzy measures and fuzzy integrals: a survey," *Fuzzy Automata and Decision Processes*, vol. 89, p. 102, 1977.
[9] S. Miyamoto, "Fuzzy Sets in Information Retrieval and Cluster Analysis," *Theory and Decision*, 1990.
[10] D. Radev, J. Otterbacher, and Z. Zhang, "CST Bank: A Corpus for the Study of Cross-document Structural Relationships," in *Proceedings of LREC 2004*, 2004.
[11] B. Baldwin and B. Carpenter, "LingPipe."
[12] H. Dalianis and E. Åström, "SweNam-A Swedish Named Entity recognizer. Its construction, training and evaluation," 2001.
[13] S. Manna, Z. Petres, and T. Gedeon, " Significant term extraction by Higher Order SVD," in *The 7th International Symposium on Applied Machine Intelligence and Informatics*, 2009.